PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-235116
(43)Date of publication of application : 13.09.1996

(51)Int.Cl. G06F 15/00
G06F 15/16

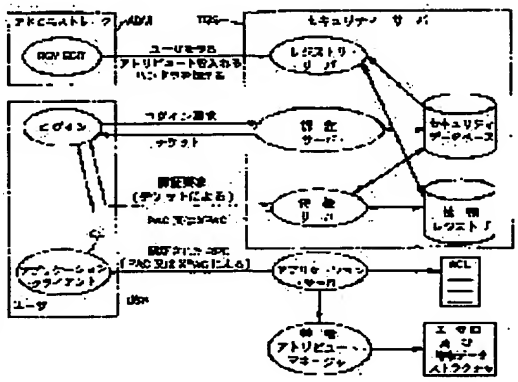
(21)Application number : 07-322822
(22)Date of filing : 12.12.1995

(71)Applicant : INTERNATL BUSINESS MACH CORP <IBM>
(72)Inventor : FORTINSKY MICHAEL S

(30)Priority
Priority number : 94 2138302 Priority date : 15.12.1994 Priority country : CA

(54) MECHANISM FOR PROVIDING SAFETY PROTECTIVE ACCESS TO EXTERNAL RESOURCE FROM DISTRIBUTED COMPUTING ENVIRONMENT

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a safe protective access to an external resource from a distributed computing environment.
SOLUTION: In a distributed computing environment in which a ticket is issued to a client from a security server TGS when the client performs access to a server, the client can access a resource on the outside of the environment upon receiving an extension certificate containing additional data from the security server TGS. The additional data give the information on the privileged attribute, etc., of the client in a format acceptable by an external server, recognized by a server which performs access to the external server, and are transmitted in the environment in a format equivalent to that of the normal ticket. The security server TGS has a registry which is extended to contain the data on the privileged attribute of the client together with the data on the structure in which the data on the privileged attribute must be given with respect to accessible external servers.



LEGAL STATUS

[Date of request for examination] 07.11.1997
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration] withdrawal
[Date of final disposal for application] 25.03.2002

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平8－235116

(43)公開日　平成８年(1996)９月13日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ	15/00	３３０	9364－5L	Ｇ０６Ｆ	15/00	３３０Ｃ
	15/16	３７０			15/16	３７０Ｎ

審査請求　未請求　請求項の数４　ＯＬ　（全 19 頁）

(21)出願番号　特願平7－322822

(22)出願日　平成７年(1995)12月12日

(31)優先権主張番号　２１３８３０２

(32)優先日　1994年12月15日

(33)優先権主張国　カナダ（ＣＡ）

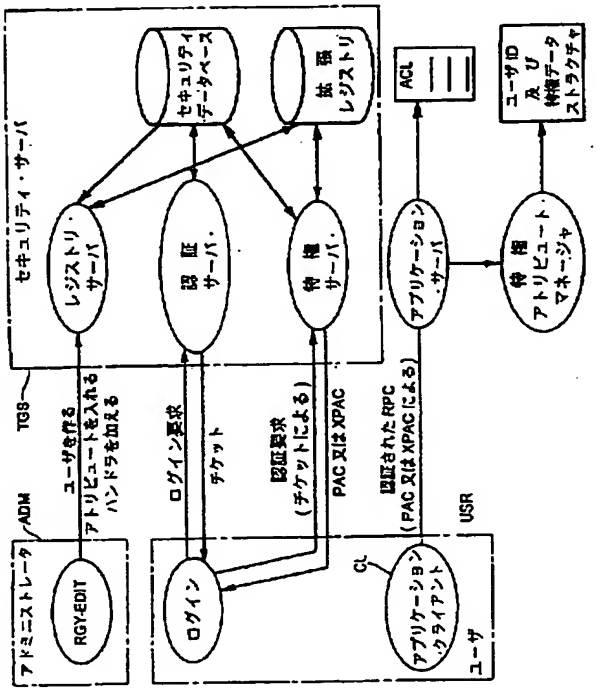
(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　ミカエル・エス・フォーティンスキー
イスラエル国ネタニア、スミランスキー、85／17　（番地なし）

(74)代理人　弁理士　合田　潔　（外２名）

(54)【発明の名称】　分散型計算環境から外部資源への安全保護アクセスを提供する機構

(57)【要約】
【課題】分散型計算環境から外部資源への安全保護アクセスを提供する
【解決手段】サーバをアクセスするためにクライアントがセキュリティ・サーバによってチケットを発行される分散型計算環境において、その環境の外部の資源へのアクセスをセキュリティ・サーバが追加データを含む拡張証明を発行することによって行う。その追加データは、クライアントの特権アトリビュート等に関する情報を、外部サーバにとって受容可能なフォーマットで与えるようにし、又、その追加データは、外部サーバへのアクセスを行うサーバによって認識され、正規のチケットと同等のフォーマットでその環境内で伝送される。セキュリティ・サーバは、アクセス可能な外部サーバに関して、クライアントの特権アトリビュートに関するデータを、そのようなデータが与えられるべきストラクチャに関するデータと共に含むよう拡張されたレジストリを有する。



【特許請求の範囲】
【請求項１】クライアント・アイデンティティ及びアトリビュートと環境における資源に関連したクライアント特権とに関するデータを含んだアトリビュート・レジストリを有するセキュリティ・サーバと、環境の外部の資源へのアクセスを行い且つ環境のセキュリティ要件とは互換性のないセキュリティ要件を有する該環境内の少なくとも１つのアプリケーション・サーバとを含むタイプの分散型計算環境にして、前記セキュリティ・サーバは前記環境内のサーバによるサービスを必要とするクライアントに対して要求に応じてチケットを発行し、前記チケットは、前記環境内のクライアントのアイデンティティ及び特権アトリビュートに関する情報を与えるように、サーバへの供給時にデコード可能であるコード化データを含む特権アトリビュート証明を有する分散型計算環境において、
前記セキュリティ・サーバは、前記外部の資源のうちの少なくとも１つに関するクライアント・アイデンティティ及び特権アトリビュートに関する追加情報と各外部の資源が前記追加情報を必要とするというストラクチャに関するデータとを含む拡張レジストリと、サーバが外部の資源へのアクセスを行うことによるサービスのためにクライアントによってリクエストされたチケットに前記追加情報を更なるコード化データとして含むための手段とを有すること、及び外部の資源へのアクセスを行うサーバは、更なるコード化データを認識するための手段と、該認識されたデータをデコードし、外部資源へのアクセスのために必要なストラクチャに該デコードされたデータを配置するための手段とを有することを特徴とする分散型計算環境。
【請求項２】前記セキュリティ・サーバ及び前記外部の資源へのアクセスを行うサーバは前記セキュリティ・サーバの前記証明における前記追加情報を含むアトリビュート・ハンドラを含み、外部の資源へのアクセスのための構造化データを必要とするサーバにおいて前記追加情報をデコード及び構造化することを特徴とする請求項１に記載の分散型計算環境。
【請求項３】前記更なるコード化データは前記環境内のクライアントの特権アトリビュートに関する結果のコード化データに続く単一のデータ・エレメントに含まれることを特徴とする請求項１に記載の分散型計算環境。
【請求項４】サーバへのアクセスを望んでいるクライアントにチケットを発行するためのセキュリティ・サーバと、環境の外部の少なくとも１つの資源をアクセスすることができる少なくとも１つのアプリケーション・サーバとを含むタイプの分散型計算環境に対する拡張にして、前記チケットはクライアントのアイデンティティ及び特権アトリビュートに関するコード化情報を含む特権アトリビュート証明を含む分散型計算環境に対する拡張機構において、
前記セキュリティ・サーバにおけるデータベースからの、クライアントのアイデンティティ及び特権アトリビュートに関する追加のコード化データ及びそのようなデータが前記環境の外部のサーバに与えられるストラクチャに関する追加のコード化データをストラクチャ内に含むべく特権アトリビュート証明が拡張されるチケットを発行するように前記セキュリティ・サーバを再構成するための手段と、
前記拡張された特権アトリビュート証明を認識するように、前記証明から前記追加データをデコードするように、及び前記外部のサーバに表示するためのデータを構造化するように、前記アプリケーション・サーバのセキュリティ・モジュールを再構成するための手段と、
を含む分散型計算環境に対する拡張機構。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、分散計算を行うコンピュータ・ネットワークにおけるセキュリティに関するものであり、更に詳しく云えば、そのようなネットワークにおいてクライアントによるアクセスを可能にすること、及びそのネットワークからアクセス可能な資源に対するチケット・ベースのセキュリティ・システムを利用するが、異なるベースのセキュリティ・システム或いは互換性のないセキュリティ・システムを利用することに関するものである。
【０００２】
【従来の技術】分散型計算システムは、分散したアプリケーション及びデータの共用を行うためにネットワークを介してコミュニケートする２つ又はそれ以上の計算機の編成を含んでいる。分散型計算システムの一例は、異種の計算機及びオペレーティング・システムに関連する分散計算をサポートするために、オープン・ソフトウエア・ファウンデーション（Open Software Foundation、以下、ＯＳＦという）によってリリースされた分散計算環境である。ＯＳＦの分散計算環境（以下、ＤＣＥという）は、マサチューセッツ工科大学（ＭＩＴ）においてその機構のアテナ（Ａｔｈｅｎａ）プロジェクトの一部として開発されたカーベロス（Ｋｅｒｂｅｒｏｓ）ネットワーク認証に基づくチケット・ベースのセキュリティ・システムを利用する。カーベロスに関する更なる詳細に対しては、そのプロジェクトの刊行物を参照することができるが、その詳細は本発明の一部分も形成するものではない。
【０００３】ＤＣＥのようなクライアント・サーバ・ネットワーク環境では、クライアント・ログイン・リクエストを検証したクライアント・アトリビュートの登録を維持するセキュリティ・サーバ又はＴＧＳ（チケット賦与サーバ）が、クライアントの特権アトリビュートの詳細を表す特権チケット賦与チケット（ＰＴＧＴ）をクライアントに発行する。しかる後、クライアントは、その

ネットワーク上のアプリケーション・サーバによるサービスを求めるリクエストによって、セキュリティ・サーバにこのチケットを与えることが可能になる。そのアプリケーション・サーバは、クライアント及びそれのセキュリティ・アトリビュートを認証する更なるチケット又はキーをそのクライアントに戻す。そのチケット又はキーは、サービスを求めるリクエストによってＤＣＥサーバに供給可能である。ＤＣＥでは、そのクライアントのアイデンティティ及びアトリビュートは、クライアントとセキュリティ及びアプリケーション・サーバとの間で送られる種々のチケットに含まれた特権アトリビュート証明（ＰＡＣ）に含まれる。

【０００４】そのようなシステムの更なる詳細は、プレンティス・ホール社（Prentice Hall, Inc）が１９９２年に発行した「ＯＳＦ　ＤＣＥの紹介（Introduction to OSF DCE）,Open Software Foundation」という文献に見ることができる。そのようなシステムでは、ＰＡＣは、サービス・リクエスト（一般には、リモート・プロシージャ・コール（ＲＰＣ））に応答して発行されたチケットにおいてセキュリティ・サーバがクライアントのＰＡＣを配置することができるように、及びクライアントによりチケットを与られるアプリケーション・サーバがリクエストを受けるべきかどうかをＰＡＣから確認できるように、ＤＣＥ環境に関連したアイデンティティ及びセキュリティ・アトリビュート（それ自体及びグループのアトリビュート・セット）を含む。

【０００５】ＤＣＥでは、クライアントのアイデンティティ及び特権がクライアントの特権アトリビュート証明（ＰＡＣ）においてサーバに送られる。ＰＡＣに含まれた情報に基づいて、サーバにより許可決定が行われる。これは、許可決定を行う時、クライアント及びサーバがＤＣＥのためにフォーマットされたそれ自体及びグループ・セットの情報のみを使用する限り十分に働く。

【０００６】異種の計算環境では、非ＤＣＥ資源、即ち、ＤＣＥ環境の外の資源に対するゲートウェイとして作用するＤＣＥサーバが存在し得る。これらの資源へのアクセスは、ＤＣＥ－ＰＡＣを理解しない非ＤＣＥアクセス制御マネージャによって制御されるであろう。その代わり、これらＡＣＬマネージャは、種々の形式のアイデンティティ及び特権情報、一般には、英数字のユーザＩＤ及びグループＩＤ、に基づいて許可決定を行うであろう。ＤＣＥ特権情報をそのようなゲートウェイに与えるＤＣＥクライアントは、それらクライアントがそれらのＰＡＣにおいてＤＣＥ特権アトリビュートを与えるだけなので非ＤＣＥ資源をアクセスすることはできないであろう。一方、それら資源と関連したアクセス制御マネージャは、非ＤＣＥ特権アトリビュートを予期し、理解する。そのようなサーバに到達するＤＣＥクライアント・リクエストは、互換性のない認証及び許可アイデンティティ方法がそのクライアント及びサーバによって使用されるため、外部資源からサービスを受けることができないであろう。

【０００７】この問題を解決するためには、クライアントの非ＤＣＥアイデンティティ及び特権をそれのＤＣＥアイデンティティと関連付けるセキュリティ機構が必要である。これは、ＤＣＥクライアントがゲートウェイ・サーバを通して外部データ及び資源にアクセスすることを可能にする。

【０００８】ＤＣＥクライアントの非ＤＣＥ特権をＤＣＥサーバに転送するための種々の方法が考えられた。それらの利点及び欠点について、簡単な検討を後述することにする。

【０００９】第１の方法によれば、アプリケーション・サーバによって要求されるすべてのクライアント特権が、セキュリティ・サーバ上に保持されたレジストリからアプリケーション・サーバによって検索される。ＤＣＥクライアントは、アプリケーション・サーバに対してチケットをリクエストし、そのチケットをアプリケーション・サーバに正規の方法で与える。クライアントのＰＡＣはＤＣＥ特権アトリビュートしか含まない。クライアントの非ＤＣＥ特権アトリビュートは、拡張アトリビュートとしてレジストリに記憶される。アプリケーション・サーバがクライアントの非ＤＣＥアトリビュートのうちのどれかを要求する場合、それは、レジストリに対する明示的なＲＰＣを作ることによってそれらをそのレジストリから検索する。これの利点及び欠点は次のようである。即ち、

利点：

（１）クライアント・コードに対して変更の必要がない。

（２）セキュリティ・サーバにおけるカーベロス・コードに対して変更の必要がない。

欠点：

（１）ＤＣＥにおいては、ＰＡＣはクライアントからサーバに特権を転送する機構である。この技法はこの原理に反するであろう。

（２）パフォーマンス。アプリケーション・サーバは、クライアント・リクエストが満足される前にレジストリを照会しなければならない（即ち、リモート・プロシージャ・コール行う）。

【００１０】第２の方法によれば、クライアントの非ＤＣＥ特権は拡張ＰＡＣ（ＸＰＡＣ）におけるそれのＤＣＥ特権と結合することである。非ＤＣＥ特権アトリビュートをＸＰＡＣに配することが可能な種々の時点がある。

【００１１】（ａ）自動的に、ＰＴＧＴが最初にクライアントによって獲得される時。これは現在のＤＣＥ－ＰＡＣ機構に類似している。現在のＤＣＥにおいては、クライアントは、それのログイン処理の間にＤＣＥグループ（即ち、それのＤＣＥ特権）の完全なセットを検索

し、グループ・リストをそれのログイン・コンテキストに配置する。その後にクライアントがＰＴＧＴをリクエストする時（ログイン・シーケンス中のその後の時点において、或いは、それがそれの第１サーバ・チケットをリクエストする時）、そのグループ・リストは新しいＰＴＧＴに対するリクエストと共に提起される。それらグループは特権サーバによって検証され、ＰＡＣの形でＰＴＧＴに配置される。この自動的特権検索の方法はクライアントの拡張特権にも適用可能である。しかし、拡張特権を組み込むことは、クライアントがログイン時にそれのすべての拡張特権を検索すること及びそれらをＰＴＧＴリクエストにおいて提起することを必要とする。それの利点及び欠点は次のようである。即ち、

利点：

（１）カーベロスＴＧＳリクエスト処理は変更を必要としない。

欠点：

（１）クライアント・ログイン・コードが変更されなければならない。

（２）クライアントはすべての特権を、それらが要求されているかどうかに関係なく、検索する。

（３）十分に基準化していない。

【００１２】（ｂ）要求に基づいて、ＰＴＧＴが最初に獲得される時。これは前の方法と類似している。しかし、拡張アトリビュートは自動的には検索されない。代わりに、クライアントが非ＤＣＥサーバをアクセスすることを知った時、すべての拡張アトリビュートが検索される。この方法はクライアントにおける何らかのインテリジェンスを必要とする。クライアントは、それがＸＰＡＣを欲しているかどうかを決定しなければならない。それの利点及び欠点は次のようである。即ち、

利点：

（１）カーベロスＴＧＳリクエスト処理は変更を必要としない。

欠点：

（１）クライアントは、それが非ＤＣＥサーバをアクセスすることを知らなければならない。

（２）クライアントはすべての特権を、それらが要求されているかどうかに関係なく検索する。

（３）十分に基準化していない。

【００１３】（ｃ）要求に基づいて、特定サーバに適合されたＰＴＧＴが獲得される時。この方法では、クライアントは特定の拡張特権だけをリクエストし（それが特定の非ＤＣＥ資源をアクセスすることを欲するため）、そして、ＸＰＡＣにおいてこれらの特権を含む新しいＰＴＧＴのためのリクエストを提起する。それの利点及び欠点は次のようである。即ち、

利点：

（１）カーベロスＴＧＳリクエスト処理は変更を必要としない。

（２）十分に基準化する。

欠点：

（１）クライアントは高度にインテリジェントでなければならない。

（２）クライアントは、それが非ＤＣＥサーバをアクセスすることを知らなければならない（いつも可能とは限らない。例えば、委任が行われている場合クライアントが最終的なターゲット・サーバについて知ることが可能でないこともある）。

（３）クライアントは、どのようなアトリビュートをＰＡＣに配置すべきかを知らなければならない。

（４）クライアントは、それが非ＤＣＥサーバをアクセスしたい時、新しいＰＴＧＴをリクエストしなければならない。

【００１４】

【発明が解決しようとする課題】本発明は第２の方法の更なる変形を使用する。本発明の目的は、計算環境においてチケット・ベースのセキュリティ・システムを実施することにある。なお、その計算環境では、資源をアクセスするためにクライアントに対して発行されるチケットに含まれた特権許可証明又は同等のデータ・エレメントは、その環境からアクセス可能な資源をアクセスするためにクライアントにとって必要なアイデンティティ及び特権データを含む必要がある場合には拡張可能であるが、通常の許可パッケージと互換性のないセキュリティ・システムを利用する。

【００１５】本発明を実施し得る例示的環境として本願を通して利用されるＤＣＥのコンテキストにおいて、これは次のような目的を達成するように設計された拡張ＰＡＣ（又は、ＸＰＡＣ）の利用を伴う。即ち、

（１）ＰＡＣに非ＤＣＥ特権アトリビュートを挿入するための機構を提供する。

（２）ＤＣＥクライアント・コードに対する変更を必要としない（それによって、ＤＣＥクライアントにおける完全な透明性を維持する）。

（３）ＸＰＡＣを使用しないすべてのＤＣＥサーバ（即ち、既存の（ＤＣＥ１．０）技法を使用して構築されたＤＣＥサーバ、及び拡張ＰＡＣを理解せず、それを期待もしない現在及び将来のＤＣＥサーバ）との相互運用性を維持する。

（４）既存のＴＧＳ及びセキュリティ実行時コードに対する修正を最小にする。

（５）必要なリモート・プロシージャ・コールの数を最小にする。

（６）新しい拡張アトリビュートをシステムに容易に追加することを可能にする。

（７）分散環境を十分に大きいものに基準化する。

【００１６】ＸＰＡＣ機構の使用はＤＣＥ環境の構成に次のような制約を課する。即ち、

（１）ＸＰＡＣを使用するサーバを含んだセルのセキュ

リティ・サーバは、ＸＰＡＣ拡張を呼び出すことができるものでなければならない。
（２）ＸＰＡＣを使用したいサーバは、ＸＰＡＣを処理できる実行時コードを含まなければならない。
【００１７】
【課題を解決するための手段】本発明によれば、クライアントは、拡張特権がそれのＰＡＣに加えられるべきであることさえ知らない。ＴＧＳがアプリケーション・サーバに対してチケットを発行する時、それは、それらを必要とするアプリケーション・サーバに対してだけ拡張アトリビュートを加える。同様に、ＴＧＳは、アプリケーション・サーバが必要とする特定のアトリビュートだけを加える（クライアントのアトリビュートすべてを加えるのとは対照的に）。利点はクライアントにおける全体的な透明性及び良好な基準化であり、欠点はＴＧＳリクエスト処理が変更を必要とすることである。
【００１８】本発明によれば、セキュリティ・サーバ及び少なくとも１つのアプリケーション・サーバを含むタイプの分散型計算環境における改良が提供される。そのセキュリティ・サーバは、クライアント・アイデンティティ及びアトリビュートとその環境における資源に関連したクライアント特権とに関するデータを含んだアトリビュート・レジストリを有する。アプリケーション・サーバは、その環境の外部の資源へのアクセスを行い且つその環境のセキュリティ要件とは互換性のないセキュリティ要件持った資源へのアクセスを行う。又、セキュリティ・サーバは、その環境内のサーバによるサービスを必要とするクライアントに対して、要求に応じてチケットを発行する。それらチケットはコード化データを含む特権アトリビュート証明を有し、その特権アトリビュート証明は、その環境内のクライアントのアイデンティティ及び特権アトリビュートに関する情報を与えるように、サーバへの供給時にデコード可能である。本発明によるこのようなセキュリティ計算環境における改良では、セキュリティ・サーバは、前記外部の資源のうちの少なくとも１つに関するクライアント・アイデンティティ及び特権アトリビュートに関する追加情報と各外部の資源がその追加情報を必要とするというストラクチャに関するデータとを含む拡張レジストリと、サーバが外部の資源へのアクセスを行うことによるサービスのためにクライアントによってリクエストされたチケットにそのような追加情報を更なるコード化データとして含むための手段とを有し、外部の資源へのアクセスを行うサーバは、更なるコード化データを認識するための手段と、そのようなデータをデコードし、その外部の資源へのアクセスのために必要なストラクチャにそれを配置するための手段とを有する。
【００１９】
【発明の実施の形態】図１は本発明を組み込んだネットワークの部分におけるセキュリティ相互作用を示し、図２はそのネットワークの関連部分の構成を示す。図２はクライアントと、ネットワークで接続された別の計算機上で走る種々のサーバとを示す。場合によっては、同じ計算機上で走るプロセスによって相異なるサーバが実行されることがあり、又その同じ計算機がＤＣＥ環境の内部及び外部の両方における資源を持つことがあり、又１つのサーバにおける多くのインスタンスが相異なる計算機上で走ることがあることを理解すべきである。図１において、セキュリティ・サーバは、ＵＳＲのようなユーザを作るために、ここではＲＧＹ－ＥＤＩＴ、即ち、ＤＣＥアトリビュート・レジストリ・エディタと呼ばれるプログラムを利用してクライアントのＤＣＥアトリビュートを入れるアドミニストレータＡＤＭと相互作用する。なお、そのユーザはアプリケーション・サーバＳＶＲにリモート・ファンクション・コールＲＰＣを発行するアプリケーション・クライアントＣＬを走らせる。
【００２０】サーバＳＶＲがＤＣＥサーバである場合、それはその内容をアクセス制御リストＡＣＬと比較するためにクライアントによって与えられる特権アトリビュート証明ＰＡＣを処理するであろう。ユーザＵＳＲがログインする時、ログイン・プロセスはセキュリティ・サーバＴＧＳにおける認証サーバにログイン・リクエストを送る。その認証サーバは、それがＤＣＥ資源へのアクセスをリクエストすることを可能にするチケットＰＴＧＴをユーザに発行する。ユーザのアプリケーション・クライアントがサーバＳＶＲの資源をアクセスする必要がある場合、それはその目的のためのチケットをセキュリティ・サーバＴＧＳにリクエストする。そのセキュリティ・サーバは、クライアントが供給するためのＰＡＣを含むサーバ・チケットをサーバＳＶＲに与える（ユーザが適正な特権を有するものと仮定して）。以下の事項はすべて正規のＤＣＥオペレーションに適用する。
【００２１】本発明によって与えられる拡張は、図２において概略的に示されたネットワークＮ１との関係において更に後述される。図２では、ＤＣＥネットワークはゲートウェイ・サーバＧＳを含み、そのゲートウェイ・サーバは非ＤＣＥサーバＲＳを、図示のような第２の非ＤＣＥネットワークＮ２によって或いは同じ計算機内に設けることによってアクセス可能である。各サーバは、それのネットワーク又はローカル及びネットワーク・オペレーティング・システムＯＳによるネットワークに接続され、各セキュリティ・サーバ及びアプリケーション・サーバは、ネットワークから受け取ったリクエストのセキュリティ事項を処理するためのセキュリティ・ルーチン・プロセスＳＲを有するであろう。
【００２２】説明される本発明の実施例の中心的な特徴は拡張ＰＡＣ又はＸＰＡＣである。ＰＡＣは、ＤＣＥクライアントに適用するＤＣＥアイデンティティ及び特権アトリビュートを含んだデータ・ストラクチャである。ＰＡＣは、カーベロス・チケットとして一般に知られた

チケットの許可データ・フィールドにおいてクライアントからサーバに送られる。許可データは許可データ・エレメントのリストより成る。ＰＡＣはこれらエレメントの１つであり、正規のＤＣＥでは、それは許可データにおける唯一のエレメントである。

【００２３】ＰＡＣは簡単にはチケットにコピーされない。それは、先ず、ピックル（ｐｉｃｋｌｅ）され、しかる後、許可データに変換される。ＰＡＣをピックルすることはＰＡＣデータ・ストラクチャを平滑化し、ネットワークを通して転送可能なフォーマットにそれらフィールドを変換することである。これは、チケットが暗号化される前に行われる。ピックルされた結果は、ピックルと呼ばれることが多い。

【００２４】チケットがＤＣＥサーバに到達する時、サーバのセキュリティ実行時モジュールは許可データからＰＡＣをアセンブルし直す責任がある。暗号化されたチケットは暗号解読され、許可データはピックルされたＰＡＣに変換され、しかる後、アンピックル（ｕｎｐｉｃｋｌｅ）されなければならない。ＰＡＣをアンピックルすることは、ピックル・プロセスのアクションを逆にすることに関連する。ＰＡＣの送信されたフォーマットはデータ・ストラクチャ・フォーマットに戻し変換される。

【００２５】ＸＰＡＣは、ＰＡＣと同じ方法でクライアントからサーバに転送される。それはピックルされ、チケットの許可データ・フィールドにおける単一の許可データ・エレメントとして送られる。許可データ・エレメントそのものは、単一のピックル（基本ＤＣＥのピックルされたＰＡＣ）ではなく、ピックルのリスト（基本ＤＣＥのピックルされたＰＡＣに続く一組の追加のピックルであり、各追加のピックルは拡張アトリビュートを表す）を保持するように拡張される。

【００２６】特権及びアイデンティティは、各セキュリティ機構が異なって定義するエンティティである。ＤＣＥクライアントのアイデンティティは、ローカル・エリア・ネットワークのような他の計算環境におけるクライアントのアイデンティティとは異なる形式で表示される。しかし、アイデンティティ及び特権を表示する方法に関係なく、本発明は、ＤＣＥクライアントがその種々のアイデンティティ及び特権のすべてをＸＰＡＣにおいて与えることを可能にする。これを達成するために次のようなオブジェクトが使用される。即ち、

特権アトリビュート・オブジェクト
特権アトリビュート・ハンドラ
特権アトリビュート・マネージャ
特権オブジェクト

これらのオブジェクトの各々については、更に詳しく後述することにする。

【００２７】ＸＰＡＣ設計における特権の基本単位は特権アトリビュート・オブジェクトである。このオブジェクトは３つの情報片、即ち、アトリビュート・タイプ、アトリビュート・エンコーディング、及びアトリビュート値を含む。アトリビュート・エンコーディングはアトリビュートをピックルに変換する方法を指定する。２つの一般的なアトリビュート・タイプ、即ち、単純アトリビュート及び複雑アトリビュートがある。単純アトリビュートは、単一のエンティティより成るアトリビュート値を含む。単純アトリビュートは、エンコーディング／デコーディング機能のデフォルト・セットを使用してエンコード及びデコード可能である。例えば、単純アトリビュートは、単一の文字ストリング（又は、単一の整数）を含むアトリビュート値を持つものでよい。アトリビュートは、文字ストリング（又は、整数）エンコーディング／デコーディングを使用してエンコード／デコードされるであろう。複雑アトリビュートは如何ように複雑であってもよい。複雑アトリビュートのアトリビュート値は文字ストリング、整数、及びバイト・ストリームの組合せを含むものでよい。複雑アトリビュートは、アトリビュート値のフォーマットの知識を持つカストマイズされた機能によってのみエンコード／デコード可能である。これらのカストマイズされた機能はアトリビュート・ハンドラ（後述のアトリビュート・ハンドラの説明参照）において与えられる。

【００２８】上述のように、アトリビュート値に含まれた情報は、そのアトリビュートが単純アトリビュートであるか或いは複雑アトリビュートであるかによって異なるであろう。単純アトリビュートに対しては、アトリビュート値は単一の情報片であり、そのアトリビュートから直接に抽出可能である。複雑アトリビュートに対しては、その値は多くの情報片を含むことがある。アトリビュート・ハンドラは、種々の情報片を抽出する方法を知る機能を提供しなければならない。

【００２９】複雑アトリビュートの例は、２つの情報片、即ち、ユーザＩＤを表す文字ストリングとグループＩＤを表す文字ストリングのリストとを含む。このアトリビュートを供給されるアトリビュート・ハンドラは、そのアトリビュートに含まれる任意の個々の情報片（例えば、ユーザＩＤ）を抽出するために使用される抽出機能を与えるであろう。

【００３０】ＤＣＥ環境において使用されるべき複雑特権アトリビュートは、そのアトリビュートを定義及び操作するために使用される一組の定義及び機能によっても達成されなければならない。特権アトリビュートに適用する定義及び機能のパッケージは特権アトリビュート・ハンドラと呼ばれる。

【００３１】特権アトリビュート・ハンドラは次のものを具体化する。即ち、（１）そのハンドラによりサポートされるアトリビュートをサーバのアトリビュート・マネージャに登録するために使用される初期設定機能、及び（２）アトリビュート・クラスについての情報を得る

ために、又はアトリビュート・クラスのインスタンスを処理及び操作するために利用可能な機能を含む機能ベクトル。

【００３２】初期設定機能はアトリビュート・クラスについての情報を戻す。これはアトリビュート・クラス識別子（名前及びＵＵＩＤ）及びアトリビュート・クラス機能ベクトルを含む。アトリビュート・クラスのための機能ベクトルは次のことを行う機能を含む。即ち、

（１）アトリビュート・クラスのインスタンスを作成する。

（２）アトリビュート・クラスを識別するＵＵＩＤを戻す。

（３）アトリビュート・クラスを識別する文字ストリング名を戻す。

（４）アトリビュート・インスタンスに含まれた情報を印刷する。

（５）アトリビュート・インスタンスをピックルする。

（６）アトリビュート・インスタンスをアンピックルする。

（７）アトリビュート・インスタンスから情報を抽出する。

（８）アトリビュート・インスタンスに置かれるべき情報に対してレジストリを照会する。

（９）アトリビュート・インスタンスに割り振られた資源を自由にする。

（１０）アトリビュート・インスタンスから以前に取り出された情報を自由にする。

【００３３】各アトリビュート・ハンドラはそれのアトリビュート・クラスのインスタンスを処理するために上記機能のそれ自身のカストマイズされたバージョンを与えるであろう。

【００３４】特権アトリビュート・ハンドラＰＡＨ（図２）は、ＴＧＳによって及びターゲット・アプリケーション・サーバＧＳによって使用される。ＴＧＳは、特権アトリビュート・ハンドラに与えられた機能を使用してレジストリからアトリビュートの種々の部分を検索し、アトリビュートを形成し、しかる後、それをピックルする（従って、それはカーベロス・チケットのＸＰＡＣ部分に配置可能である）。ターゲット・アプリケーション・サーバはハンドラ機能を使用して、ＸＰＡＣから抽出後にアトリビュートをアンピックルしそしてそのアトリビュートから特定の情報を抽出する。アトリビュート・ハンドラ機能は、アプリケーション・サーバ・コード又はＴＧＳコードによって直接に呼び出されることはないことに留意すべきである。それらは、ＸＰＡＣセキュリティ実行時ＡＰＩ（アプリケーション・プログラミング・インターフェース）及びＸＰＡＣアプリケーション・サーバＡＰＩの下で呼び出される。これは、アプリケーション・サーバ及びＴＧＳロジックをアトリビュート・ハンドラ機能から切り離す。

【００３５】一般に、特権アトリビュート・ハンドラは（静的に又は動的リンク・ライブラリ（ＤＬＬ）を介して）アプリケーション・サーバ・コードとリンクされる。チケット許可セキュリティ・サーバの場合には、例外が生じる。ＴＧＳの場合、アトリビュート・ハンドラは、セキュリティ・サーバが初期設定する時、動的にサーバに付加される。これは、ＴＧＳがＴＧＳコードを再コンパイル及び再リンクする必要なしにアトリビュート・ハンドラを追加又は削除することを可能にするために行われる。アトリビュート・ハンドラがセキュリティ・アトリビュートを操作するため、セキュリティアドミニストレータだけがセキュリティ・サーバにアトリビュート・ハンドラを加えることを許可されなければならない。

【００３６】特権アトリビュート・マネージャＰＡＭ（図２）は、どのクラスの複雑アトリビュートがサーバに対して定義されるかを表す情報を記憶するコンポーネントである。各サーバは１つの特権アトリビュート・マネージャ・コンポーネントを含む。

【００３７】サーバが複雑アトリビュートを操作できる前に、それはアトリビュート及びそれの関連のハンドラをアトリビュート・マネージャに登録しなければならない。（アトリビュート登録は、サーバ初期設定時に行うことが可能である）。初期設定機能のアドレスをアトリビュート・マネージャに送ることによって、アトリビュートが登録される。この機能はアトリビュート・ハンドラによって広められる。アトリビュート・マネージャは、アトリビュート・クラスに関する次のような情報、即ち、アトリビュート・クラスＵＵＩＤ、アトリビュート・クラス名、及びアトリビュート・クラス機能ベクトル、を取得しそして記憶するために初期設定機能を使用する。

【００３８】ＵＵＩＤ及び名前はアトリビュート・クラスを識別し、機能ベクトルはアトリビュート・クラスのインスタンスを操作するに必要な機能を与える。アトリビュート・マネージャはアトリビュート・ハンドラからこの情報をすべて取得し、それを内部のルックアップ・テーブルに記憶する。

【００３９】アトリビュートがサーバによって処理されるべき場合、そのサーバは、先ず、そのアトリビュートが単純アトリビュートであるか又は複雑アトリビュートであるかを決定しなければならない。そのアトリビュートが単純アトリビュートである場合、デフォルト・ルーチンが使用可能である。そのアトリビュートが複雑アトリビュートである場合、サーバはアトリビュート・マネージャを照会してその複雑アトリビュートが登録されているかどうかを決定する。それが肯定される場合、アトリビュート・マネージャは、その後そのアトリビュートを処理するために使用可能な機能ベクトルをサーバに戻す。その複雑アトリビュートが登録されていない場合、

アトリビュート・マネージャはエラー・ステータスを戻し、サーバはそのアトリビュートを無視しなければならない。

【００４０】特権オブジェクトは特権アトリビュートのコンテナである。ＸＰＡＣは、ＤＣＥアトリビュート及び非ＤＣＥアトリビュートを含む特権オブジェクトである。例えば、ＸＰＡＣはＤＣＥ部分及び１つのローカル・ホスト・セキュリティ・サブシステム特権アトリビュートを含むことができる。特権オブジェクトの観点から、それが含む特権アトリビュートは不透明なデータ・オブジェクトである。特権オブジェクトは任意の数の特権アトリビュート・オブジェクトを含むことができる。

【００４１】図２は、ＸＰＡＣ拡張を使用するクライアント・サーバ交換に関連の主要なコンポーネントを示す。ＤＣＥサーバをアクセスすることを欲するＤＣＥクライアントはＤＣＥ　ＴＧＳからそのサーバに対するチケットをリクエストする。そのＴＧＳは、ターゲット・サーバが非ＤＣＥ特権機構と関連のサーバであるかどうかを決定するためにそれの拡張レジストリを調べる。非ＤＣＥ特権機構と関連したターゲット・サーバは、クライアントがＰＡＣの代わりにＸＰＡＣ（ＤＣＥ特権に加えて非ＤＣＥ特権も含む）を与えることを要求する。ターゲット・サーバがＸＰＡＣを要求する場合、ＴＧＳは拡張レジストリからそのクライアントの適当な拡張アトリビュートを得て、ＸＰＡＣを形成する。

【００４２】結局、クライアントはネットワークＮ１を介してサーバにサービス・チケットを与える。サーバは、ＰＡＣに対するプロシージャと同じプロシージャを呼び出すことによってクライアントのＸＰＡＣに対するポインタを得る。そこで、サーバは、後述のＡＰＩを使用してＸＰＡＣから種々の拡張アトリビュートを抽出することができる。

【００４３】ＤＣＥクライアントの観点からは、変化は全くない。非ＤＣＥ資源へのアクセスをリクエストするクライアントは、その資源へのアクセスを制御するＤＣＥサーバにチケットをリクエストする。クライアントはそれのＰＴＧＴをＴＧＳに送り、それに応答してサーバ・チケットを受け取る。

【００４４】クライアントが受け取ったチケットは、正規のＤＣＥ－ＰＡＣではなくＸＰＡＣを含む。これはクライアントにとって透明である。結局、クライアントがターゲット・サーバを呼び出す時、それはＸＰＡＣを含むサーバ・チケットを送る。

【００４５】アドミニストレータがクライアント及びターゲット・サーバ自体に対する拡張レジストリにおいて拡張アトリビュート情報を構成した後にしか、ＸＰＡＣは形成可能ではなく、使用可能でもない。拡張特権アトリビュートを使用するターゲット・サーバに対して、アドミニストレータは、ターゲット・サーバが使用する各拡張アトリビュートに対するサーバ特権機構レコードを定義しなければならない。サーバ特権機構は、サーバへのアクセスに必要な特権アトリビュートのセット及び各特権アトリビュートに適用するオプショナル追加データを含むようにターゲット・サーバのレジストリを論理的に拡張する。

【００４６】そのセットにおける各特権アトリビュートはＵＵＩＤによって指定される。このＵＵＩＤは、レジストリから直接に検索されそしてクライアントのＸＰＡＣに置かれるべき単一の単純アトリビュートのＵＵＩＤであるか、或いはアトリビュート・ハンドラが形成しそしてＸＰＡＣに挿入する複雑アトリビュートのＵＵＩＤであろう。ＸＰＡＣに配置されるべき特権アトリビュートが何らかの方法で修飾されなければならない場合、この目的のためには追加のデータを使用することが可能である。

【００４７】特権機構及び追加のデータの使用を、次のような例によって最も良く説明することができる。

【００４８】サーバ１は、クライアントがそのサーバの非ＤＣＥ資源へのアクセスを得る前に、そのクライアントがそれのアクセス・コードをＸＰＡＣにおけるそれのローカル・ホストに与えることを必要とする。サーバ１は計算機Ａ、Ｂ、Ｃ、Ｄ、及びＥに存在する。計算機Ａ、Ｂ、及びＣにおけるサーバ１のインスタンスはタイプＡ１のアトリビュートを使用し、一方、計算機Ｄ及びＥにおけるサーバ１のインスタンスはタイプＢ１のアトリビュートを使用する。タイプＡ１及びＢ１のアトリビュートは両方とも単純アトリビュートである。従って、必要なデータ・ストラクチャ、即ち、単純なデフォルト・ストラクチャが知られるように、ＸＰＡＣは単純アトリビュートとして識別されるそのクライアントのタイプＡ１及びＢ１アトリビュートを含まなければならない。

【００４９】サーバ２は、外部資源へのゲートウェイ・アクセスを与えるサーバである。これらの資源をアクセスするために、クライアントは、ユーザ・プロファイル全体を含む（ユーザＩＤ、グループ・リスト、及び他のセキュリティ・データを含む）複雑アトリビュートを与えなければならない。単純アトリビュートのリストとして個々のアトリビュートすべてを指定する代わりに、複雑な特権アトリビュートＡ２が定義される。アトリビュートＡ２のインスタンスはそれの値フィールドにユーザ・プロファイルを含む。Ａ２は、Ａ２のアトリビュート・ハンドラがセキュリティ・サーバ及びターゲット・サーバの両方において導入される場合のみ使用可能である。Ａ２のハンドラは、ＸＰＡＣに及びＸＰＡＣからユーザ・プロファイルを封入し及び抽出する方法を知っているコードである。アドミニストレータは、サーバ１及びサーバ２に対するレジストリにおける次表に示すデータを指定するであろう。即ち、

【表１】

サーバ自体	必要な特権機構	追加データ
サーバ１	Ａ１	Ａ、Ｂ、Ｃ
	Ｂ１	Ｄ、Ｅ
サーバ２	Ａ２	なし

【００５０】追加データとラベル付けされたフィールドは、ＸＰＡＣに配置されるべき情報を更に修飾するために使用可能なオプショナル・エントリである。このフィールドにおける情報のフォーマット及び意味は、それが適用するアトリビュートのコンテキストにおいてのみ理解可能である。上記の表１において、追加データ「Ａ、Ｂ、Ｃ」はアトリビュートＡ１のコンテキストにおいてはローカル計算機Ａ、Ｂ、及びＣを意味するように解される。

【００５１】ＲＧＹ－ＥＤＩＴを使用するアドミニストレータは必要な特権機構を文字ストリングとして入れる。これらはＵＵＩＤに変換される。追加データは、それが適用するアトリビュート・タイプにとって意味のあるフォーマットで入れられる。例えば、追加データが計算機のリストである場合、それも文字ストリングのリストとして入れることが可能である。これらの文字ストリングは内部的にＵＵＩＤに変換可能である。又、アドミニストレータは、上記ターゲット・サーバをアクセスする各クライアントに対して必要な拡張特権アトリビュートをすべて入れなければならないであろう。クライアントの主要なエントリは次のように思われる。即ち、クライアント自体：

ベースＤＣＥアトリビュート
拡張アトリビュート　－－－＞Ａ１－Ａ－値１
Ａ１－Ｂ－値２
Ａ１－Ｃ－値３
Ｂ１－Ｄ－値４
Ｂ１－Ｅ－値５

【００５２】拡張特権アトリビュートをセキュリティ・レジストリ・データベースＤＢに加えるための機構が必要である。適当な機構の一例は、オープン・ソフトウエア・ファウンデーションから入手可能なＤＣＥ　ＲＦＣ　６．０　において提案された「拡張レジストリ・アトリビュート（ＥＲＡ）」である。以下では、この必要な機構をＥＲＡと呼ぶことにする。そのＥＲＡ機構は、ＤＣＥアドミニストレータによって呼び出され、拡張サーバ及びクライアント・アトリビュートＥＲＡをサーバ及びクライアント・レジストリ・エントリＤＢ（図１）に加える。このＥＲＡ機構は、外部ＡＰＩを使用して、及びＤＣＥ管理ユーティリティ（例えば、ＲＧＹ－ＥＤＩＴ）を介してアクセス可能でなければならない。同様に、ＴＧＳは、レジストリからの拡張特権アトリビュートを検索するためにＥＲＡに対するアクセスを持つであろう。

【００５３】無修正のＤＣＥでは、ＴＧＳがサーバ・チケットに対するクライアント・リクエストを受け取る時、それは受信したＰＴＧＴ（カーベロス許可データ・フィールドにＰＡＣを含む）を暗号解読及びデコードし、ＰＴＧＴから宛先サーバ・チケットに許可データを盲目的にコピーし、しかる後、その結果のチケットを再エンコード及び再暗号化する。

【００５４】ＸＰＡＣ設計は２つのロケーション、即ち、セキュリティ・サーバ初期設定及びＴＧＳリクエスト処理、におけるセキュリティ・サーバに変更を強いる。セキュリティ・サーバが先ずスタートする時、それは余分の構成ファイルを読み取らなければならない。このファイルはファイル名のリストを含む。各ファイル名は複雑な特権アトリビュートに対するアトリビュート・ハンドラを含むファイルを表す。セキュリティ・サーバは、各アトリビュート・ハンドラに含まれた機能を動的にロードする。アトリビュート・ハンドラＰＡＨを動的にロードすることによって、新しいアトリビュートが、再コンパイル又は再リンクの必要なしにそのシステムに追加可能である。ハンドラを含むファイル名を構成ファイルに追加すること、ＴＧＳを停止させること、しかる後、ＴＧＳを再開始させることによって、（新しい、更新されたハンドラのリストがロードされ）、新しいアトリビュートが追加可能である。これの代替え方法の１つは、構成ファイルの管理がＲＧＹ－ＥＤＩＴの枠組に統合される場合、ＴＧＳを停止させることなく構成ファイルを更新しそして新しいハンドラをロードするである。

【００５５】ＴＧＳがアトリビュート・ハンドラをロードする時、それは特権アトリビュート・マネージャＰＡＭに、そのハンドラを登録する。その後、ＴＧＳがハンドラの機能をアクセスする必要がある時、アトリビュート・マネージャはそのアクセスを行うであろう。

【００５６】ＴＧＳに対する第２の変更では、ＰＡＣを保持した入力許可データがＸＰＡＣを保持した出力許可データに変えられるように、ＴＧＳリクエスト処理がわずかに修正される。サーバ・チケットに対するクライアント・リクエストがＴＧＳに到達する時、現在のＤＣＥにおけるように、ＴＧＳリクエスト処理は正規のデコーディング及び暗号解読オペレーションから始まる。しかし、ＰＴＧＴからサーバ・チケットに許可データをコピーする前に、ＴＧＳは２ステップのルックアップ・プロシージャを開始する。それの第１ステップでは、ＴＧＳは、ターゲット・サーバが何れかの非ＤＣＥサーバ特権機構と関連しているかどうかを決定するためにＥＲＡ機構（上記参照）を使用する。そのようなサーバ特権機構が存在しない場合、ＴＧＳは、正規のＴＧＳリクエスト処理を使用してターゲット・サーバ・チケットを発行する。しかし、何れかの非ＤＣＥサーバ特権機構がターゲット・サーバに適用する場合、ＴＧＳはそのルックアップの第２ステップを開始する。

【００５７】ルックアップの第２ステップは、各非ＤＣＥサーバ特権機構に対して適用するクライアントの拡張アトリビュートを検索することに関連する。各特権機構は、ターゲット・サーバに与えられなければならない拡張特権アトリビュートのクラスを識別する（即ち、特権機構ＵＵＩＤはアトリビュート・クラスＵＵＩＤに等価である）。各機構に対して、ＴＧＳは、先ず、そのクラスのアトリビュートに対するハンドラが存在するかどうかを決定するために特権アトリビュート・マネージャを照会する。それが存在する場合、特権アトリビュート・マネージャは、そのアトリビュート・クラスに適用する機能ベクトルを戻す。ＴＧＳはこのベクトルを使用して、そのクライアントに属し且つそのサーバにおいて適用するアトリビュートを検索する。その機能ベクトルは、再びそのアトリビュートをピックルするために使用され、ＴＧＳはそのピックルを許可データ・フィールド（基本ＤＣＥ特権を含んでいる）に付加し、そしてサーバ・チケットはそのクライアントに戻される。

【００５８】アトリビュート・ハンドラが存在しない場合、ＴＧＳは、必要なアトリビュートが単純アトリビュートであると仮定し、クライアントのレジストリ・エントリからそのアトリビュートを検索するためにＥＲＡ機構を使用する。そのアトリビュートはデフォルト・ピックル・ルーチンを使用してピックルされ、ＴＧＳはサーバ・チケットにおける許可データ・フィールドにそのピックルを付加し、そしてサーバ・チケットはクライアントに戻される。

【００５９】拡張特権を持った許可データ・フィールドのフォーマットは次のように見える。即ち、

許可データ・カテゴリ（ＯＳＦ－ＤＣＥ）
長さ（基本及び特別ピックルを含む）
内容－－－＞ 基本ＤＣＥピックル
アトリビュート１ピックル
アトリビュート２ピックル
・・・
アトリビュートＮピックル

【００６０】拡張特権アトリビュートを必要とするアプリケーション・サーバは、それがＸＰＡＣを扱わなければならないことを知っている。そのサーバは、それが必要とする拡張アトリビュートのタイプも知っている。ＸＰＡＣに含まれた情報をサーバが使用するためには、そのサーバは、それが使用するアトリビュートのタイプを登録しなければならず、そしてＸＰＡＣに含まれた拡張アトリビュートから所望の情報を取り出さなければならない。

【００６１】サーバが拡張アトリビュート・タイプを登録する時、それは、セキュリティ実行時モジュールがそのアトリビュート・タイプを処理するに必要な情報を与える。この情報はそのアトリビュートに対する初期設定機能のアドレスである。サーバがアトリビュートを登録する時、そのサーバに対する特権アトリビュート・マネージャは初期設定機能を使用してそのアトリビュートに対するクラス情報を取得し、そして記憶する。この情報はアトリビュート・クラス指示子（ＵＵＩＤ）、アトリビュート・クラス・ストリング名、及びそのクラスからアトリビュートのインスタンスを操作するために使用されるルーチンを含む機能ベクトルを含む。ターゲット・サーバの特権アトリビュート・マネージャは、ＴＧＳが記憶するのと同じ方法でこの情報を内部ルックアップ・テーブルに記憶する（図１参照）。

【００６２】ＸＰＡＣを含むチケットがサーバに到達する時、そのサーバの実行時セキュリティ・モジュールはカーベロス許可データを分析し、ＸＰＡＣ特権オブジェクトを形成する。その分析中、それは、先ず、ＤＣＥ特権を抽出する。拡張（即ち、ＤＣＥピックルに付加された追加のピックル）が存在する場合、それは各拡張のクラスを調べ、そのクラスのための機能ベクトルに対する特権アトリビュート・マネージャを照会する。特権アトリビュート・マネージャがそのクラスを登録している場合、その機能ベクトルは戻され、ベクトルのアンピックル・ルーチンが呼び出され、そしてアンピックルされたアトリビュートが特権オブジェクトに加えられる。アトリビュート・マネージャがそのクラスに対するハンドラを登録していない場合、アトリビュートは無視される。アプリケーション・サーバは、前述の外部ＡＰＩの１つを呼び出すこと及び所望のアトリビュートを抽出することによって、拡張特権をアクセスすることができる。ＤＣＥ１．０サーバは、ＸＰＡＣに含まれた拡張特権を無視するであろう。それらはＸＰＡＣをＤＣＥ１．０ＰＡＣのように扱うであろう。

【００６３】ＸＰＡＣに対応する特権オブジェクトは次のような形式を持つであろう。即ち、

基本ＤＣＥ部分
拡張ＵＵＩＤ
アトリビュートの数
アトリビュート１－－－－－－＞Ａ１フィールド
アトリビュート２－－－－－－＞Ａ２フィールド

【００６４】拡張ＵＵＩＤは、セキュリティ実行時モジュールにとって良く知られたＵＵＩＤである。それは、基本ＤＣＥ部分に続く拡張アトリビュートの存在を表す。ＵＵＩＤの不存在は、古いスタイルのＰＡＣが処理されようとしていることを表し、従って、セキュリティ実行時モジュールは、何れの拡張も処理しようとはしないであろう。

【００６５】次の事項は拡張ＰＡＣ処理に関連した主要ステップである。

（１）ＴＧＳが始動し、すべての特別の特権アトリビュート・ハンドラをロードする。

（２）サーバＳが始動し、それが認識するすべての特別のアトリビュートを登録する。

（３）クライアントＣがＴＧＳにサーバＳのチケットをリクエストする。
（４）ＴＧＳは、サーバＳが何らかの拡張特権アトリビュートを必要としているかどうかをチェックする。
（５）それが肯定される場合、ＴＧＳはそれらアトリビュートのインスタンスに対してクライアントＣのレジストリ・エントリを照会し、それらをサーバ・チケットに挿入する。
（６）ＴＧＳがサーバＳに対するチケットをクライアントＣに戻す。
（７）クライアントＣがサーバＳにリクエストを送り、サーバＳにチケットを渡す。
（８）サーバＳのセキュリティ実行時モジュールがそのチケットからＸＰＡＣを抽出する。
（９）サーバＳはＸＰＡＣから種々のアトリビュートを明瞭にリクエストし、それらを必要に応じて使用する。
【００６６】正規のＤＣＥでは、他のセルにおけるサーバをアクセスすることを望んでいる１つのセルにおけるクライアントは、それら２つのセルにおけるＤＣＥセキュリティ・アドミニストレータによって信頼関係が形成されている場合、そのように行うことができる。サーバのセルにおけるＴＧＳがそのクライアントに対するサービス・チケットを、起点クライアントのＰＡＣを使用して発行することができるので、これは可能である。
【００６７】しかし、１つのセルにおけるクライアントが他のセルにおけるサーバをアクセスすることを望み且つそのサーバがそのクライアントの拡張アトリビュートを必要とする場合、そのモデルは更に複雑になる。そのサーバがそのクライアントの拡張アトリビュートを必要とすることを知っている唯一のエンティティはそのサーバのセルにおけるＴＧＳである。このＴＧＳは、そのクライアントに属する拡張アトリビュートを検索できなければならない。クライアント・セル・セキュリティ・サーバが拡張特権アトリビュートをサポートすることを仮定することはできない。事実、そのサーバのセルは、外部のクライアントをそのセルに相互登録するための機構を与えなければならず、そしてこれら外部のクライアント・エントリに拡張アトリビュートを加えなければならない。
【００６８】この設計は、ＥＲＡ機構が外部クライアント自体及びそれらのアトリビュートを１つのセルに相互登録するための機構を与えることを仮定している。このような機構の場合、ＸＰＡＣを必要とする外部セルにおいて生じたリクエストは次のような方法で処理されるであろう。即ち、
（１）サーバのセルにおけるＴＧＳが始動し、すべての特別な特権アトリビュート・ハンドラをロードする。
（２）サーバＳが始動し、それが認識するすべての特別アトリビュートを登録する。
（３）そのサーバのセルにおけるアドミニストレータがそのセルにクライアントＣを相互登録し、サーバＳに適用するそのクライアントの拡張アトリビュートを加える。
（４）クライアントＣがそれ自身のＴＧＳにサーバＳのチケットをリクエストする。
（５）マルチ・セル相互作用が開始され、その結果、サーバＳのチケットに対するリクエストがそのサーバのＴＧＳに行われる。
（６）このＴＧＳは、サーバＳが何らかの拡張特権アトリビュートを必要としているかどうかをチェックする。
（７）それが肯定される場合、ＴＧＳはそれらアトリビュートのインスタンスに対して外部クライアント自体Ｃのための相互登録されたエントリを照会し、それらをサーバ・チケットに挿入する。
（８）ＴＧＳがサーバＳに対するチケットをクライアントＣに戻す。
（９）クライアントＣがサーバＳにリクエストを送り、サーバＳにチケットを渡す。
（１０）サーバＳのセキュリティ実行時モジュールがそのチケットからＸＰＡＣを抽出する。
（１１）サーバＳはＸＰＡＣから種々のアトリビュートを明瞭にリクエストし、それらを必要に応じて使用する。
【００６９】この実施例は、ＸＰＡＣを認識しないすべてのサーバとの相互運用性を維持する。これは可能なことである。それは、サーバがアドミニストレータによって適正に登録されると仮定すると、ＸＰＡＣを認識するサーバだけがそれらを受け取り、ＸＰＡＣを認識しないサーバはサーバ・チケットにおいてそれらを受け取らないためである。たとえ、アドミニストレータがサーバを、ＸＰＡＣを認識するものとして不正確に登録しても、サーバはＸＰＡＣにおける拡張を無視するであろうし、ＸＰＡＣをＤＣＥ　１．０　ＰＡＣとして扱うであろう。
【００７０】この実施例はＤＣＥセキュリティ・サービスにおいて包含するための新しいＡＰＩを提案する。その設計は既存の基本ＤＣＥコードに対する非常にわずかな変更を必要とするだけであり、基本ＤＣＥに対する次のような変更でもって基本ＤＣＥに組込可能である。即ち、
（１）特権アトリビュート・ハンドラをロードするためのセキュリティ・サーバ始動コードにおけるフック。
（２）アトリビュートをチケットに加えるためのＴＧＳ処理コードにおけるフック。
（３）入力ＸＰＡＣをアンパックするためのセキュリティ実行時モジュールにおけるフック。
【００７１】アクセスされるべき特定の資源にとって特有の特権アトリビュート処理コードのすべては、動的にロードされる（ＴＧＳによって）か、或いは静的にリンクされる（アプリケーション・サーバによって）。それ

は、アトリビュート・ハンドラ・ルーチンに対して指定されたフォーマットに適合する必要がある。拡張アトリビュートを利用することを望んでいるアプリケーション・サーバは新しいＡＰＩを呼び出す。

【００７２】本願において開示されるＸＰＡＣ設計は、ＰＡＣに加えられるべきＤＣＥクライアントの非ＤＣＥ特権アトリビュートのための機構を与える。その設計の特性は次のように要約するすることができる。即ち、

（１）クライアント側のコードに対するコード変更は必要なく、その機構はクライアントにとって透明である。

（２）クライアントは、ターゲット・サーバの性質に関する明瞭な知識を維持する必要がない。

（３）ＸＰＡＣを認識しないサーバは影響されない。

（４）セキュリティ・サーバのＴＧＳに対して、及びＸＰＡＣを使用することを欲するアプリケーション・サーバのセキュリティ実行時モジュールに対して、わずかな修正が必要なだけである。

（５）追加のリモート・プロシージャ・コールは導入されない。

（６）新しいアトリビュート・ハンドラをロードすることによって、追加の拡張アトリビュートが適応可能である。

（７）ＸＰＡＣに配置された拡張アトリビュートは特定のターゲット・サーバへのアクセスのために必要なものだけである。

【００７３】Ａ．セキュリティ・サーバのためのＡＰＩ

（ａ）特権アトリビュート・ハンドラをロードする

パラメータ：なし

概説：このＡＰＩは、それが処理できる特権アトリビュートのリストをロードするためにＴＧＳによって呼び出される。

高レベル・フロー：

・＜アトリビュート・ハンドラ・モジュールのファイル名＞を構成ファイルから読み取る。

・アトリビュート・ハンドラの初期設定ルーチン（登録機能）をそのモジュールから動的にロードする。

・特権アトリビュート・ハンドラを登録するためのルーチン（下記参照）を呼出し、初期設定ルーチンのアドレスを入力として渡す。

【００７４】（ｂ）許可データを付加する

パラメータ：

入力

クライアント	ＴＧＳリクエストを行うクライアントの名前
サーバ	ＴＧＳリクエストのターゲットであるサーバの名前

入力－出力

許可データ	クライアントの許可データ－新しい特権が付加される

概説：これは、ＴＧＳが入力のＴＧＳリクエストをデコードし且つ暗号解読した後、ＴＧＳによって呼び出される内部機能である。そのリクエストにおけるターゲット・サーバが、クライアントが特別な非ＤＣＥ特権与えることを必要とするものである場合、この機能はこれらの特権を検索し、許可データに含まれた既存の特権にそれらを付加する。

高レベル・フロー：

・入力の許可データを調べ、ＤＣＥ許可データである第１許可データ・エレメントを決定する。

・クライアント及びサーバ名をＵＵＩＤに変換する。

・ターゲット・アプリケーション・サーバが非ＤＣＥ特権機構を使用するかどうかを決定する。

・サーバによって要求される各特権機構に対して、

－アトリビュートが登録されたハンドラ機能を有するかどうかを決定する

－アトリビュートがハンドラを有する場合、

－アトリビュートのインスタンスを作成するためにそれの作成機能を呼び出す

－クライアントの拡張レジストリ・エントリからアトリビュート値を検索するためにそれの照会レジストリ機能を呼び出す

－アトリビュートが登録されたハンドラを持たない場合、

－実際のアトリビュートに関してレジストリを照会する

－アトリビュートをピックルする

－許可データに保持されたピックルのセットにそのピックルを付加する

【００７５】Ｂ．セキュリティ実行時機能

（ａ）拡張ＰＡＣを形成する

パラメータ：

入力

許可データ	一組の付加ピックルより成るカーベロス許可データ

出力

ＰＡＣ	ＸＰＡＣのアンピックル・バージョン

概説：この内部機能はアプリケーション・サーバのセキュリティ実行時モジュールによって呼び出される。それは１つ又は複数個のピックルされた特権項目を含むカーベロス許可データを変換し、その特権をアンピックルし、ＸＰＡＣを形成する。この機能は既存の許可データ・ツー・ＰＡＣ処理を置換する。

高レベル・フロー：

・カーベロス許可データを一組のピックルされたアトリ

ビュートに変換する
・ＤＣＥ部分をアンピックルする
・各追加のピックルに対して、
－アトリビュートをアンピックルする
－アトリビュートをＸＰＡＣ特権オブジェクトに挿入する
・完成したＸＰＡＣを戻す
【００７６】実行時機能は、入力ＸＰＡＣに対してスペースを割り振るため及びこのスペースが最早必要ない時にはこのスペースを自由にするためにも、特権アトリビュート・マネージャに対するルックアップ・テーブルを初期設定しそしてアクセスするためにも、そのようなテーブルが最早必要ない時にはそのテーブルと関連した資源を開放するためにも、そして特権アトリビュート・マネージャを登録及び登録解除するためにも与えられる。
【００７７】Ｃ．アプリケーション・サーバに対するＡＰＩ
（ａ）特権アトリビュート・ハンドラを登録する
パラメータ：
　入力
　　登録情報　　特権アトリビュートに対する初期設定機能のアドレスを含む不透明なデータに対するポインタ
　出力
　　アトリビュート・クラス　　登録されたアトリビュートを識別するＵＵＩＤ
概説：このＡＰＩは、それが認識しそして処理する特権アトリビュートを登録するために、サーバによって呼び出される。
高レベル・フロー：
・登録情報から初期設定機能を抽出する
・アトリビュートを登録するために特権アトリビュート・マネージャを呼びだし、その機能ベクトル、クラスＵＵＩＤ、及びクラス名を戻す
・クラスＵＵＩＤを起呼者に戻す
【００７８】（ｂ）特権アトリビュート・ハンドラを登録解除する
パラメータ：
　入力
　　アトリビュート・クラス　　登録解除されるべきアトリビュート・クラスを識別するＵＵＩＤ
概説：このＡＰＩは、それが前に登録した特権アトリビュートを登録解除するためにサーバによって呼び出される。
高レベル・フロー：
・アトリビュートを登録解除するために特権アトリビュート・マネージャを呼び出す
【００７９】（ｃ）カーソルを初期設定する
パラメータ：
　出力
　　カーソル　　初期設定されたカーソルに対するポインタ
概説：このＡＰＩは特権アトリビュート・オペレーションにおけるその後の使用のためにカーソルを初期設定する
高レベル・フロー：
カーソル・オブジェクトを割り振る
カーソルを初期設定する
【００８０】（ｄ）カーソルをリセットする
パラメータ：
　入力・出力
　　カーソル　　カーソル・オブジェクトに対するポインタ
概説：このＡＰＩは既存のカーソルをリセットする。これは特権アトリビュートの照会を再開させるために行われる。
高レベル・フロー：
・カーソル・オブジェクトをリセットする
【００８１】（ｅ）カーソルを削除する
パラメータ：
　出力
　　カーソル　　カーソル・オブジェクトに対するポインタ
概説：このＡＰＩは既存のカーソルによって保持された資源を自由にする
高レベル・フロー：
・カーソル・オブジェクトを削除する
【００８２】（ｆ）ＸＰＡＣから基本ＤＣＥ－ＰＡＣを抽出する
パラメータ：
　入力
　　特権　　クライアントのＸＰＡＣに対するポインタ
　出力
　　ＤＣＥ特権　　ＸＰＡＣから取り出された基本ＤＣＥ－ＰＡＣに対するポインタ
概説：このＡＰＩはＸＰＡＣから基本ＤＣＥ－ＰＡＣを抽出する。出力はＸＰＡＣに含まれたＤＣＥ－ＰＡＣのコピーである
高レベル・フロー：
・ＤＣＥ－ＰＡＣのコピーに対してメモリを割り振る
・ＤＣＥ－ＰＡＣを新たに割り振られたメモリにコピーする
【００８３】（ｇ）基本ＤＣＥ－ＰＡＣと関連の資源を自由にする

パラメータ：
入力
特権　自由にされるべき資源を有する基本ＤＣＥ－ＰＡＣ
概説：このＡＰＩは、先行のＡＰＩに対する呼出しを介して得られた基本ＤＣＥ－ＰＡＣと関連した資源を自由にする
高レベル・フロー：
・ＤＣＥ－ＰＡＣを自由にする
【００８４】（ｈ）拡張ＰＡＣと関連の資源を自由にする
パラメータ：
入力
特権　自由にされるべき資源を有する拡張ＰＡＣ
概説：このＡＰＩは拡張ＰＡＣと関連した資源を自由にする。
高レベル・フロー：
・ＸＰＡＣの動的に割り振られた部分を自由にする
・ＸＰＡＣの残りを自由にする
【００８５】（ｉ）特権アトリビュートを抽出する
パラメータ：
入力
特権　クライアントのＸＰＡＣを対するポインタ
アトリビュート・タイプ　ＸＰＡＣから取り出されるべき特権アトリビュートのタイプ
入力・出力
カーソル　カーソル・オブジェクトに対するポインタ
出力
アトリビュート　取り出された特権アトリビュートに対するポインタ
概説：このＡＰＩはＸＰＡＣから特権アトリビュートを抽出する。アトリビュート・タイプ入力パラメータが指定される場合、ＸＰＡＣはその指定されたタイプに適合したアトリビュートを見つけるために走査される。適合した特権アトリビュートに関するサーチはカーソル位置から開始する。アトリビュート・タイプ入力パラメータが「ＮＵＬＬ」にセットされる場合、次の特権アトリビュート（カーソルの後）は戻される。抽出が成功すると、カーソル位置はその取り出された特権アトリビュートを参照するために更新される。
高レベル・フロー：
・送られたＰＡＣがＸＰＡＣであることをチェックする。
・カーソル位置で開始する
・カーソルが過去の最後のアトリビュートを指す場合、エラーを戻す
・次のようにループする
－次のアトリビュートを得る
－アトリビュート・タイプが指定される場合、アトリビュート・タイプに関して一致するかどうかをチェックする
－カーソル位置を更新する
－一致が見つかるまで或いはリストの終わりまでループする
【００８６】（ｊ）特権アトリビュートのインスタンスを作成する
パラメータ：
入力
アトリビュート・タイプ　作成されるべき特権アトリビュートのタイプ
出力
アトリビュート　新たに作成されたアトリビュートに対するポインタ
概説：このＡＰＩは特定のクラスの特権アトリビュートのインスタンスを作成する。そのクラスの一般的なインスタンスが作成され、インスタンス特定値がその後の機能呼出しによって満たされたまま残される。この機能は、クライアントの証書に置かれるべきＸＰＡＣを形成する時、セキュリティ・サーバによって呼び出される。又、それは、それが入力クライアント証書を使用してＸＰＡＣを形成する時、アプリケーション・サーバのセキュリティ実行時モジュールによっても呼び出される。
高レベル・フロー：
・アトリビュート・クラスが登録されているかどうかをチェックする
・アトリビュート・インスタンスに対してメモリを割り振る
・後で満たされるべきすべてのインスタンス・フィールド（特定フィールド）を残して非インスタンス特定データでもってインスタンスを初期設定する
【００８７】（ｋ）アトリビュートと関連の資源を自由にする
パラメータ：
入力
アトリビュート　自由にされるべき資源を有するアトリビュート
概説：このＡＰＩは特権アトリビュートと関連した資源を自由にする。
高レベル・フロー：
・アトリビュート・クラスが登録されているかどうかをチェックする
・アトリビュートのアトリビュート－クラス－特定部分を自由にする
・アトリビュートと関連した資源の残りを自由にする
【００８８】（ｌ）特権アトリビュートのタイプを抽出する

パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
出力
アトリビュート・タイプ　特権アトリビュートのタイプ
概説：このＡＰＩは特定の特権アトリビュートのタイプを戻す。「アトリビュート・タイプ」出力パラメータが起呼者によって割り振られる。
高レベル・フロー：
・特権アトリビュートのタイプ（即ち、クラス）を表すＵＵＩＤを戻す
【００８９】（ｍ）特権アトリビュートから値を抽出する
パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
基準　複雑アトリビュートに対して、これはそのアトリビュートのどの部分が戻されるべきかを指定する
入力・出力
カーソル　カーソル・オブジェクトに対するポインタ
出力
アトリビュート値　特権アトリビュートに含まれた値
概説：このＡＰＩは特権アトリビュートに含まれた値を戻す。そのアトリビュートが単純アトリビュートである場合、基準及びカーソル入力は無視される。そのアトリビュートが複雑アトリビュートである場合、基準はどの値が戻されるべきかを指定する。アトリビュートが所望の基準の複数のインスタンスを含む場合、カーソルは次のインスタンスを戻すために使用される。戻されたアトリビュート値によって使用されるメモリはこの機能によって割り振られ、後続のＡＰＩに対する呼出しでもって自由にされなければならない。
高レベル・フロー：
・単純アトリビュートである場合、それの値を戻す
・複雑アトリビュートである場合、それが処理可能なものであるかどうかを知るためにルックアップ・テーブルをチェックする
・それが処理可能なものである場合、それの抽出機能を呼び出す
【００９０】（ｎ）アトリビュート値によって使用される資源を自由にする
パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
入力・出力
アトリビュート値　自由にされるべきアトリビュート値
概説：このＡＰＩはアトリビュート値によって使用される資源を自由にする
高レベル・フロー：
・単純アトリビュートの値である場合、それを自由にする
・複雑アトリビュートの値である場合、それが処理可能なものであるかどうかを知るためにルックアップ・テーブルをチェックする
・それが処理可能なものである場合、それの自由機能を呼び出す
【００９１】Ｄ．アトリビュート・ハンドラＡＰＩ
以下の機能はアトリビュート・ハンドラによって与えられる。アトリビュート・ハンドラは機能ポインタのベクトルであり、従って、それらの機能の実際の名前は重要ではない。そのベクトルが作成される時、それは、この項で説明することを実施する機能に対するポインタを与える。
（ａ）作成
パラメータ：
出力
アトリビュート　アトリビュート・ハンドラによって定義されたタイプの特権アトリビュートの新たに作成されたインスタンスに対するポインタ
概説：そのハンドラのタイプの特権アトリビュートのインスタンスを作成する
高レベル・フロー：
・アトリビュートの新しいインスタンスを割り振る
・新しいインスタンスを指すように「アトリビュート」出力パラメータをセットする
【００９２】（ｂ）タイプ
パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
出力
アトリビュート・タイプ　アトリビュートのタイプを指定するＵＵＩＤ
概説：起呼者によって割り振られた「アトリビュート・タイプ」出力パラメータを、アトリビュートのタイプを指定するＵＵＩＤに等しくセットする。「アトリビュート」入力パラメータは「ＮＵＬＬ」であってもよく、これはアトリビュート・クラスのタイプが（特定のアトリ

ビュートに含まれるタイプとは対照的に）出力として望ましいことを表す。
高レベル・フロー：
・アトリビュートが正しいクラスのもの（又は、ＮＵＬＬ）であることをチェックする
・アトリビュート・クラスに対するアトリビュート・タイプに「アトリビュート・タイプ」出力パラメータをセットする
【００９３】（ｃ）名前
パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
出力
アトリビュート名　アトリビュートの名前の文字ストリング表示
概説：アトリビュート・クラスの名前を「アトリビュート名」パラメータにコピーする。「アトリビュート」入力パラメータは「ＮＵＬＬ」であってもよい。これは、アトリビュート・クラスの名前が（特定のアトリビュートの名前とは対照的に）出力として望ましいことを表す。その名前がコピーされるバッファは起呼者によって割り振られる。
高レベル・フロー：
・アトリビュートが正しいクラスのもの（又は、ＮＵＬＬ）であることをチェックする
・「アトリビュート名」出力アーギュメントとして送られたバッファにアトリビュート・クラスのストリング名をコピーする。
【００９４】（ｄ）プリント・アトリビュート
パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
概説：アトリビュートに含まれた情報をプリントする
高レベル・フロー：
・アトリビュートが正しいクラスのものであることをチェックする
・アトリビュートに含まれた情報をプリントする
【００９５】（ｅ）エンコード
パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
出力
エンコード・アトリビュート　アトリビュートのエンコード（即ち、ピックルされた）バージョンに対するポインタ
概説：供給されたアトリビュートをピックルする
高レベル・フロー：
・アトリビュートが正しいクラスのものであることをチェックする
・ピックルを保持するために必要なメモリを計算し、割り振る。
・アトリビュートをピックルする
【００９６】（ｆ）デコード
パラメータ：
入力
アトリビュート　情報のデコード（即ち、アンピックルされた）バージョンを満たされる特権アトリビュートに対するポインタ
エンコード・アトリビュート　特権アトリビュートのエンコード・バージョンに対するポインタ
概説：
供給されたピックルをアンピックルする
高レベル・フロー：
・アトリビュートが正しいクラスのものであることをチェックする
・ピックルをアンピックルする
・アンピックルされた情報でもってアトリビュートを満たす
【００９７】（ｇ）抽出
パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
基準　抽出すべきものを指定するハンドラ特有の基準
入力・出力
カーソル　アトリビュートにおける情報を横断するために使用されるカーソルに対するポインタ
出力
アトリビュート値　アトリビュートから抽出された所望の情報に対するポインタ
概説：アトリビュートから特定情報を抽出する
注意：その抽出された情報は抽出機能によって割り振られ、そしてそれはアトリビュート値を自由にする機能に対する呼出しでもって自由にされなければならない。
高レベル・フロー：
・「カーソル」がＮＵＬＬである場合、アトリビュートに含まれた情報の始めにおいてサーチが開始する。
・それ以外の場合は、開始ポイントとしてカーソルを使用する
・入力基準に基づいてアトリビュートをサーチする
・基準が満たされる場合、「アトリビュート値」を保持するためのメモリを割り振り、情報を「アトリビュート

値」出力にコピーする
【００９８】（ｈ）照会レジストリ
パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
概説：アトリビュートに配置されるべき情報に関してレジストリを照会し、アトリビュートを満たす
高レベル・フロー：
・アトリビュートが正しいクラスのものであることをチェックする
・アトリビュート値を得るためにレジストリに対する必要な呼出しを行う
・レジストリによって戻された情報を保持するためのメモリを割り振る
・戻された情報でもってアトリビュート値を満たす
【００９９】（ｉ）アトリビュートを自由にする
パラメータ：
入力
アトリビュート　特権アトリビュートに対するポインタ
概説：アトリビュートのインスタンスに割り振られた資源を自由にする
高レベル・フロー：
・アトリビュートに含まれたデータと関連した資源を自由にする
・アトリビュートそのものと関連した資源を自由にする
【０１００】（ｊ）アトリビュート値を自由にする
パラメータ：
入力
アトリビュート値　アトリビュートから抽出された情報に対するポインタ
概説：アトリビュートのインスタンスから抽出された情報に割り振られた資源を自由にする
高レベル・フロー：
・アトリビュート値と関連した資源を自由にする
【０１０１】上記の機能は、それぞれ、正常終了又は１つ或いは複数個のエラー状態を表すために値を戻すであろう。
【０１０２】上記の説明はＤＣＥへの本発明の適用に対して特別の関連を持っているけれども、それが、公認のサーバ・アクセスに対して同様の許可プロシージャを有する他の分散型計算環境においても適用可能であることは明らかであろう。
【０１０３】まとめとして、本発明の構成に関して以下の事項を開示する。
（１）クライアント・アイデンティティ及びアトリビュートと環境における資源に関連したクライアント特権とに関するデータを含んだアトリビュート・レジストリを有するセキュリティ・サーバと、環境の外部の資源へのアクセスを行い且つ環境のセキュリティ要件とは互換性のないセキュリティ要件を有する該環境内の少なくとも１つのアプリケーション・サーバとを含むタイプの分散型計算環境にして、前記セキュリティ・サーバは前記環境内のサーバによるサービスを必要とするクライアントに対して要求に応じてチケットを発行し、前記チケットは、前記環境内のクライアントのアイデンティティ及び特権アトリビュートに関する情報を与えるように、サーバへの供給時にデコード可能であるコード化データを含む特権アトリビュート証明を有する分散型計算環境において、前記セキュリティ・サーバは、前記外部の資源のうちの少なくとも１つに関するクライアント・アイデンティティ及び特権アトリビュートに関する追加情報と各外部の資源が前記追加情報を必要とするというストラクチャに関するデータとを含む拡張レジストリと、サーバが外部の資源へのアクセスを行うことによるサービスのためにクライアントによってリクエストされたチケットに前記追加情報を更なるコード化データとして含むための手段とを有すること、及び外部の資源へのアクセスを行うサーバは、更なるコード化データを認識するための手段と、該認識されたデータをデコードし、外部資源へのアクセスのために必要なストラクチャに該デコードされたデータを配置するための手段とを有することを特徴とする分散型計算環境。
（２）前記セキュリティ・サーバ及び前記外部の資源へのアクセスを行うサーバは前記セキュリティ・サーバの前記証明における前記追加情報を含むアトリビュート・ハンドラを含み、外部の資源へのアクセスのための構造化データを必要とするサーバにおいて前記追加情報をデコード及び構造化することを特徴とする上記（１）に記載の分散型計算環境。
（３）前記更なるコード化データは前記環境内のクライアントの特権アトリビュートに関する結果のコード化データに続く単一のデータ・エレメントに含まれることを特徴とする上記（１）に記載の分散型計算環境。
（４）サーバへのアクセスを望んでいるクライアントにチケットを発行するためのセキュリティ・サーバと、環境の外部の少なくとも１つの資源をアクセスすることができる少なくとも１つのアプリケーション・サーバとを含むタイプの分散型計算環境に対する拡張にして、前記チケットはクライアントのアイデンティティ及び特権アトリビュートに関するコード化情報を含む特権アトリビュート証明を含む分散型計算環境に対する拡張機構において、前記セキュリティ・サーバにおけるデータベースからの、クライアントのアイデンティティ及び特権アトリビュートに関する追加のコード化データ及びそのようなデータが前記環境の外部のサーバに与えられるストラクチャに関する追加のコード化データをストラクチャ内に含むべく特権アトリビュート証明が拡張されるチケットを発行するように前記セキュリティ・サーバを再構成

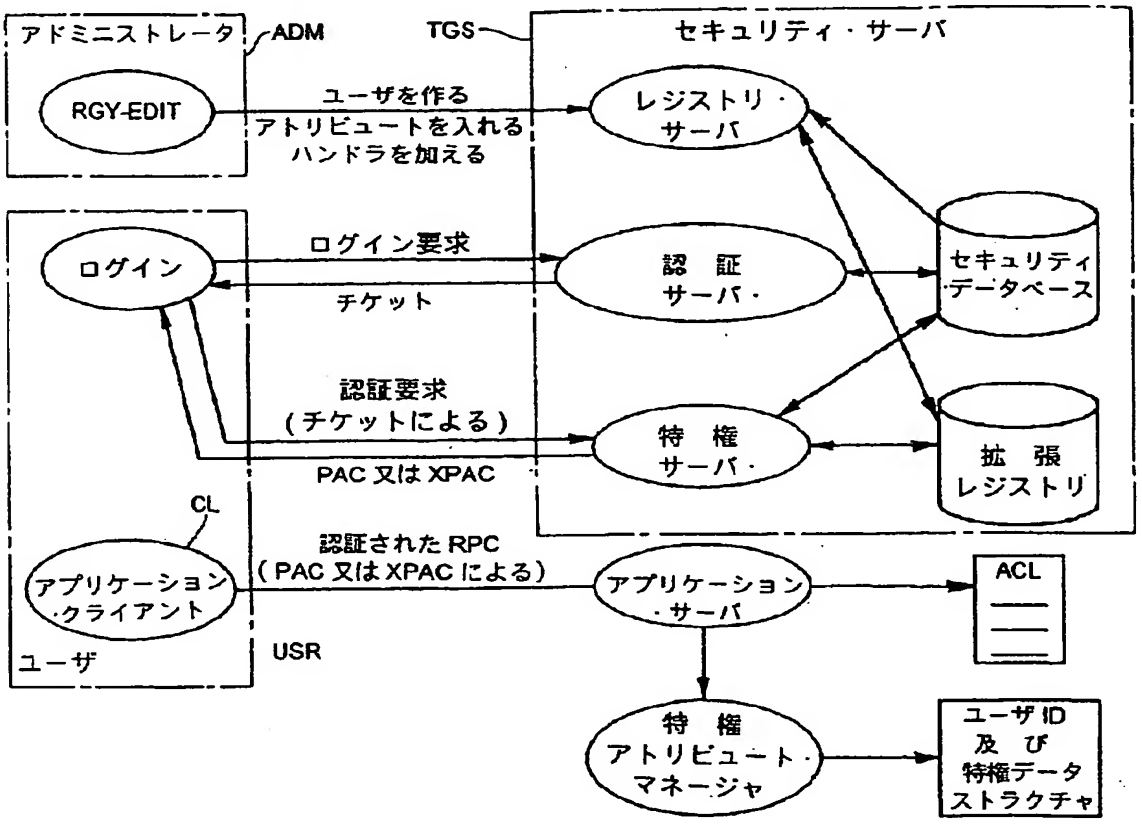
するための手段と、前記拡張された特権アトリビュート証明を認識するように、前記証明から前記追加データをデコードするように、及び前記外部のサーバに表示するためのデータを構造化するように、前記アプリケーション・サーバのセキュリティ・モジュールを再構成するための手段と、を含む分散型計算環境に対する拡張機構。

【図面の簡単な説明】

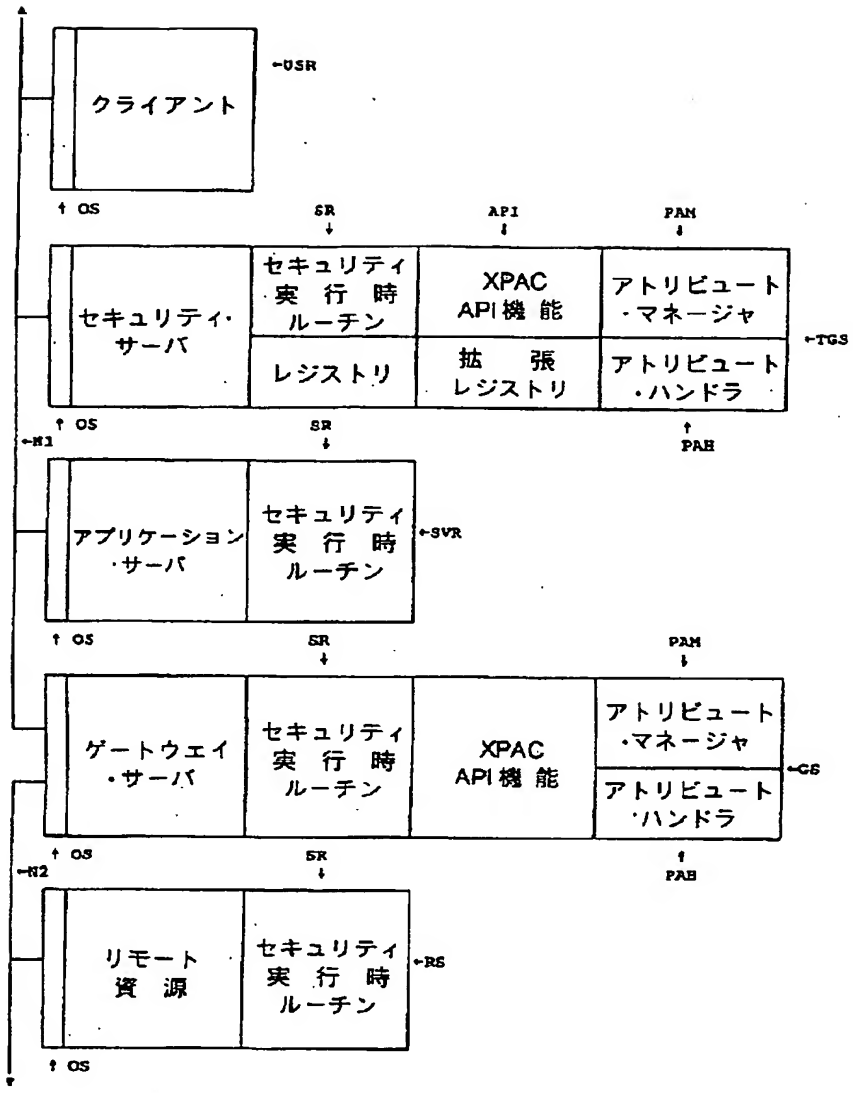
【図１】アドミニストレータ、セキュリティ・サーバ、及びユーザの間の相互作用を概略的に示す。

【図２】ネットワークの相対的部分の概略表示である。

【図１】



【図２】


クライアント
←USR
↑ OS
SR
API
PAM
セキュリティ・サーバ
セキュリティ実行時ルーチン
XPAC API機能
アトリビュート・マネージャ
レジストリ
拡張レジストリ
アトリビュート・ハンドラ
←TGS
↑ OS
SR
↑ PAH
←N1
アプリケーション・サーバ
セキュリティ実行時ルーチン
←SVR
↑ OS
SR
PAM
ゲートウエイ・サーバ
セキュリティ実行時ルーチン
XPAC API機能
アトリビュート・マネージャ
アトリビュート・ハンドラ
←GS
↑ OS
SR
↑ PAH
←N2
リモート資源
セキュリティ実行時ルーチン
←RS
↑ OS